Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Mrak

Aplikacija za dihalno vadbo na mobilnem telefonu

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: doc. dr. Matija Marolt

Ljubljana, 2016

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

V okviru diplomskega dela razvijte mobilno aplikacijo, ki bo omogočala izvajanje dihalnih vaj s tehniko dihanja skozi priprte ustnice, ki lahko uporabniku pomaga pri lajšanju nekaterih zdravstvenih težav. Aplikacija naj deluje na principu predvajanja zvočnih datotek s pomočjo dihalnih vaj in naj uporabnika motivira pri izvajanju vaj v daljšem časovnem obdobju. Deluje naj v okoljih iOS in Android.

Za nasvete in strokovno pomoč pri izdelavi diplomskega dela se zahvaljujem mentorju doc. dr. Matiji Maroltu. Za vso podporo in pomoč v času študija se zahvaljujem svojim najbližjim, brez katerih mi vsekakor ne bi uspelo.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
PLB	pursed lip breathing	izdih skozi priprte ustnice
XML	Extensible Markup Language	razširljiv označevalni jezik
SQL	Structured Query Language	strukturirani povpraševalni jezik
HTML	HyperText Markup Language	jezik za označevanje besedila, namenjen izdelavi spletnih strani

Povzetek

Naslov: Aplikacija za dihalno vadbo na mobilnem telefonu

Avtor: Luka Mrak

Tehnika dihanja skozi priprte ustnice je način izvajanja dihalnih vaj, ki se uporablja za lajšanje različnih zdravstvenih težav. V okviru diplomskega dela smo se zato odločili za razvoj mobilne aplikacije, ki je namenjena predvajanju zvočnih datotek, uporabnik pa z izvajanjem dihalnih vaj nadzira njihovo predvajanje. Da aplikacija pravilno deluje, mora uporabnik v svojo mobilno napravo vključiti slušalke z vgrajenim mikrofonom, v nasprotnem primeru namreč detekcija dihanja, in s tem posledično naša aplikacija, ne deluje optimalno. Aplikacijo smo izdelali v okolju Unity za operacijska sistema Android in iOS. Ker se operacijska sistema med seboj močno razlikujeta, smo aplikacijo ustrezno prilagodili tem razlikam.

Ključne besede: Unity, dihalna vadba, predvajalnik zvočnih datotek, mobilna aplikacija.

Abstract

Title: Pursed lip breathing exercises on a mobile phone

Author: Luka Mrak

PLB breathing technique is a method of breathing exercises used as a treatment for easing different medical conditions. The goal of this thesis is the development of a mobile application, designed to playback audio files. Users will be controlling the playing of audio files with their breathing exercises. In order for the application to function properly, the user will have to plug in his/her headset, otherwise breathing detection and, consequently, our application will not work optimally. We have developed this application using Unity for both, Android and iOS operating systems. We have modified our application accordingly to the many differences between the two operating systems.

Keywords: Unity, breathing exercises, audio player, mobile application.

Poglavje 1

Uvod

Uporabniki mobilnih naprav si z uporabo mobilnih aplikacij skušajo olajšati oz. izboljšati svoj vsakdan, tudi na področju zdravja. Težave vse pogosteje želijo lajšati v udobju svojega doma, s pomočjo mobilnih aplikacij.

Mobilno zdravstvo (angl. Mobile Health, krajše mHealth) je hitro rastoče področje, katerega primarni namen je bodisi diagnostika bodisi zdravljenje uporabnika. Mobilne aplikacije so zmožne brezžičnega prejemanja in oddajanja podatkov, omogočajo podporo multimedijskih interaktivnih vsebin in s tem prinašajo edinstveno možnost za pripravo in vključevanje pacientov v proces njihovega zdravljenja [21].

Eden od načinov lajšanja različnih zdravstvenih težav (npr. astma, kronična obstruktivna pljučna bolezen, simptom oteženega dihanja oz. dispneja) je tudi tehnika dihanja skozi priprte ustnice (angl. Pursed lip breathing, krajše PLB). Pozitiven učinek navedene tehnike je v preprečevanju kolapsa dihalnih poti, lajšanju dihanja in sprostitvi oziroma pomiritvi uporabnika [15]. PLB je danes priporočen vsem bolnikom z astmo pred in med astmatičnim napadom in vsem osebam, ki težko dihajo ali imajo premalo energije.

V okviru diplomskega dela smo se zato odločili za razvoj mobilne aplikacije, s pomočjo katere bo uporabnik motiviran k pogostejšemu izvajanju tehnike dihalnih vaj PLB. S pomočjo izdelane aplikacije in slušalk z vgrajenim mikrofonom bo uporabnik izvajal tehniko PLB tako, da bo skozi priprte

ustnice pihal v mikrofon, aplikacija pa bo merila dolžine uporabnikovih izdihov. Namen aplikacije je, da si uporabnik z rednim izvajanjem dihalnih vaj podaljša izdih.

Cilji naše aplikacije so, da uporabniku ponudi izbiro zvočnih datotek prisotnih na napravi, omogoči predvajanje zvočne datoteke z izvajanjem dihalnih vaj preko mikrofonskega vhoda, v dveh načinih delovanja, prikazuje rezultate dihalnih vaj in jih shranjuje v XML datoteko, poleg navedenega pa tudi opozarja uporabnika naj med izvajanjem vaj pije vodo (asinhrono vodenje). Nenazadnje je cilj naše aplikacije tudi v njenem zaklepanju na serijsko številko.

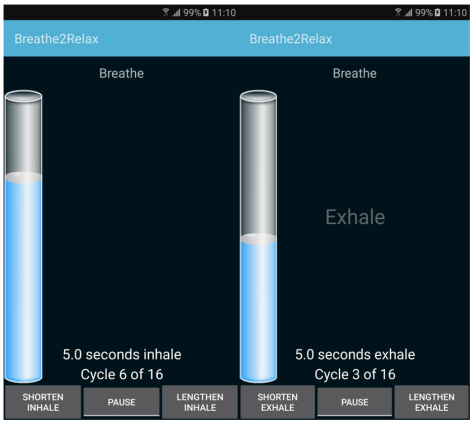
Cilj diplomskega dela je izdelati aplikacijo za predvajanje zvočnih datotek z nadzorom preko mikrofonskega vhoda v okolju Unity, za operacijska sistema Android in iOS.

Na začetku diplomskega dela bomo predstavili že obstoječe aplikacije s tega področja in tehnologije, ki smo jih uporabili za izdelavo aplikacije. Nato bomo opisali postopek izdelave aplikacije, predstavili njene posamezne dele in način uporabe.

1.1 Pregled področja

Trenutno dostopne aplikacije za izvajanje dihalnih vaj lahko razdelimo v dve skupini. V prvo skupino sodijo aplikacije, ki ne merijo uporabnikovega vdiha in izdiha, ampak uporabnika vodijo skozi dihalne vaje zgolj na podlagi nastavitev, ki jih uporabnik lahko spreminja. Takšne so na primer naslednje aplikacije:

- **Breathe2Relax** [5] uporabnika vodi skozi vadbo z vizualnimi in zvočnimi napotki, za dodatno sprostitev pa v ozadju predvaja sproščujočo glasbo. Poleg tega omogoča nastavljanje predvidene dolžine vdiha in izdiha med vadbo, kot je prikazano na sliki 1.1.



Slika 1.1: Prikaz zaslona aplikacije Breathe2Relax, levo za vdih in desno za izdih.

- **Pranayama** [19] uporabnika vodi skozi dihalno vadbo z vizualnimi napotki in uporabo animacij, za dodatno sprostitev pa v ozadju predvaja sproščujočo glasbo. Uporabniku omogoča tudi vpogled v napredek njegove dihalne vadbe in spreminjanje predvidene dolžine vdiha in izdiha med vadbo, kot prikazuje slika 1.2.

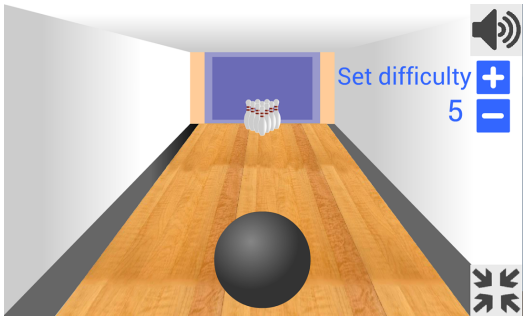
Slika 1.2: Levo prikaz aplikacije Pranayama med vadbo, desno prikaz nastavitev.

- **Pacedbreathing** [16] poleg vizualnih in zvočnih napotkov uporabnika vodi tudi preko vibracij mobilne naprave. Uporabnik lahko med dihalno vadbo nastavi dolžino njegovega predvidenega vdiha, izdiha in časa med njima, kot je prikazano na sliki 1.3.

Slika 1.3: Prikaz aplikacije Pacedbreathing, levo prikaz izdiha, desno prikaz vdiha.

V drugo skupino sodijo aplikacije, ki uporabnika vodijo glede na njegov vdih oz. izdih, ki ju zaznavajo s pomočjo aktivnosti na mikrofonu. Ločimo jih lahko v dve podskupini, glede na način izvajanja dihalnih vaj.

- V prvo podskupino sodijo aplikacije, kjer uporabnik poskuša doseči čim daljši izdih **z enim poskusom**. Primer je aplikacija **Bowling** [**4**]. Gre za igro, v kateri uporabnik nastavi težavnost, ki predstavlja dolžino oz. čas enega izdiha, ki je potreben za podrtje kegljev. Višjo težavnost kot nastavi, daljši izdih (čas) je potreben za dosego cilja, kot prikazuje slika 1.4.



Slika 1.4: Začetni zaslon aplikacije Bowling.

- V drugo podskupino lahko uvrstimo tiste aplikacije, kjer uporabnik poskuša doseči čim daljši **povprečni izdih** v daljši časovni periodi. Njegov napredek je seštevek vseh dolžin njegovih izdihov. Primer takšne aplikacije je na primer **Swimmer** [23]. Gre za igro, v kateri uporabnik nastavi težavnost, ki predstavlja potreben povprečni čas v določeni časovni periodi za dosego cilja. Cilj igre je v tem, da uporabnik s ponavljanjem izdihov skuša prehiteti ostale navidezne tekmovalce v plavanju, kot je prikazano na sliki 1.5.

Slika 1.5: Začetni zaslon aplikacije Swimmer.

Poglavje 2

Uporabljene tehnologije

V tem poglavju diplomskega dela bomo predstavili tehnologije, ki smo jih uporabili za izdelavo naše aplikacije.

2.1 Unity

Unity je igralni pogon, ki ga je razvilo podjetje Unity Technologies, primarno pa je namenjen izdelavi iger. Unity je bil prvič predstavljen leta 2005, zgolj za operacijski sistem OS X, danes pa je možen razvoj tako na operacijskih sistemih OS X kot tudi na operacijskih sistemih Windows [27].

2.1.1 Lastnosti

Ena najboljših lastnosti uporabe orodja Unity je njegova podpora za razvoj na različnih platformah, ki so prikazane na sliki 2.1. Razvijalcu namreč omogočajo, da ob izvozu projekta izbere eno izmed naslednjih platform:

- **igralne konzole:** Nintendo 3DS, Wii U, PlayStation 4, PlayStation Vita, Xbox One, Xbox 360;
- **namizne platforme:** Mac OS, Windows, Linux/Steam OS;

- **platforme za obogateno in navidezno resničnost:** Oculus Rift, Google Cardboard, Steam VR, Gear VR, Playstation VR, Microsoft Hololens;
- **mobilne platforme:** Android, iOS, Windows Phone, Tizen;
- **spletne platforme:** WebGL;
- **pametni TV sprejemniki:** tvOS, Samsung SMART TV, Android TV.

Slika 2.1: Podprte platforme [29].

Na voljo so štirje različni paketi programske opreme Unity [31], in sicer Personal, Plus, Pro in Enterprise. Unity Personal je edini paket, ki je uporabniku na voljo brezplačno. Vsi štirje paketi programske opreme vsebujejo vse funkcionalnosti pogona, vendar je uporaba Personal in Plus verzije pogona dovoljena le tistim uporabnikom, katerih prihodek v posameznem letu ne preseže določenega bruto prihodka. Primerjava med paketi pokaže, da pri uporabi paketa Personal uporabnik nima možnosti zamenjave začetne animacije (angl. splash screen), medtem ko lahko pri ostalih paketih začetno animacijo izklopi ali doda svojo. Poleg navedenega paket Personal podpira manjše število istočasnih igralcev, saj lahko igro sočasno igra največ 20 igralcev, medtem ko imajo drugi paketi višjo omejitev, in sicer 50 oz. več igralcev. Med paketi programske opreme so še druge razlike, vendar te niso bile ključne za izdelavo naše aplikacije.

Glede na potrebe naše aplikacije smo se odločili za uporabo paketa Unity Personal, ki ponuja podporo razvoja na vseh platformah, uporabljamo lahko vse funkcije pogona, poleg tega pa nudi tudi konstantne nadgradnje programske opreme.

2.1.2 Programski jeziki

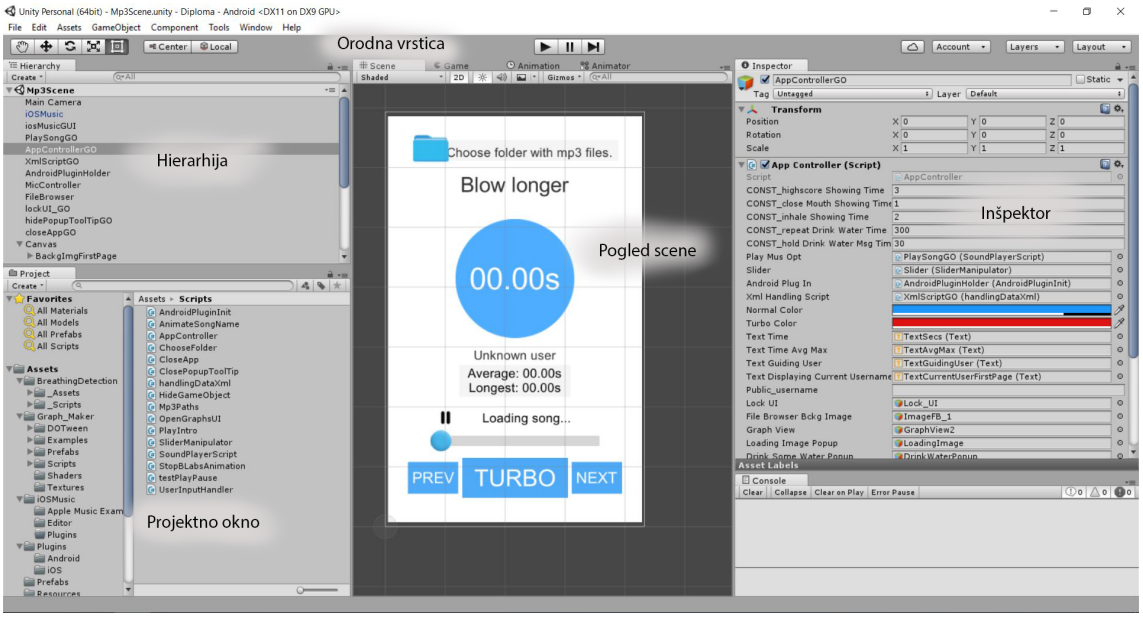
Programska oprema Unity podpira tri programske jezike, in sicer UnityScript (Javascript za Unity), Boo in C#. V okviru diplomskega dela smo uprabljali programski jezik C# s podmnožico platforme Microsoft .NET 2.0 (angl. Microsoft .NET 2.0 Subset).

2.1.3 Uporabljena verzija

Razvoj naše aplikacije smo začeli v verziji 5.2.4, ki je bila izdana dne 16. 12. 2015. Ob razvoju naše aplikacije smo imeli, tako kot številni drugi uporabniki, težave z zvokom v aplikacijah na platformi Android. Ko je mobilna naprava, na kateri smo poganjali izdelano aplikacijo, dobila obvestilo (npr. zvonenje mobilne naprave ob dohodnem klicu ali besedilnem sporočilu), je zvok v naši aplikaciji, narejeni z orodjem Unity, prenehal delovati. Ostali zvoki mobilne naprave so delovali nemoteno, napako v naši aplikaciji pa je bilo mogoče odpraviti le s ponovnim zagonom naše aplikacije, ki je nemoteno delovala, vse dokler nismo ponovno prejeli obvestila. Enake težave so imeli tudi drugi uporabniki različnih verzij, kot je razvidno tudi iz njihovega uradnega foruma [7]. Zato smo se odločili za nadgradnjo orodja Unity na verzijo 5.4.0, ki je bila izdana dne 28. 7. 2016 in ni vsebovala navedene težave.

2.1.4 Uporabniški vmesnik

Uporabniški vmesnik orodja Unity je prikazan na sliki 2.2.



Slika 2.2: Orodje Unity v uporabi.

Sestavljen je iz naslednjih komponent [12]:

- **Projektno okno (angl. Project Window)**, ki prikazuje našo zbirko sredstev (angl. Asset), ki so nam na voljo za uporabo v našem projektu. Kadar uvozimo sredstva v naš projekt, se slednji pojavijo tukaj.
- **Pogled scene (angl. Scene View)**, ki nam omogoča navigacijo in urejanje naše scene. Ta pogled lahko našo sceno prikazuje v 2D in 3D perspektivi, odvisno od tipa projekta, s katerim delamo.
- **Hierarhija (angl. Hierarchy Window)** je tekstovno predstavljena hierarhija vsakega objekta v sceni. Vsak element v sceni je del hierarhije, tako da sta ta dva okna (pogled scene in hierarhija) tesno povezana. Hierarhija nam pove, kako so objekti hierarhično povezani drug z drugim.
- **Inšpektor (angl. Inspector Window)** nam omogoča ogled in urejanje vseh lastnosti trenutno izbranega objekta. Ker imajo različni tipi

objektov različne lastnosti, se postavitev (angl. Layout) in vsebina okna inšpektorja razlikujeta v odvisnosti od objekta.

- **Orodna vrstica (angl. Toolbar)** zagotavlja dostop do najbolj osnovnih in hkrati najpomembnejših komponent za delo. Na levi strani imamo osnovna orodja za manipuliranje s sceno in objekti v njej, na sredini pa gumbe za pričetek, pavzo in korak kontrole. Gumbi na desni strani omogočajo dostop do vsebin, ki nam jih ponuja orodje Unity v oblaku (angl. Unity Cloud Services), sledi gumb s povezavo z našim računom Unity, ki mu sledijo nastavitve vidnosti plasti ter nazadnje še meni za urejanje plasti.

2.1.5 Asset Store

Spletna trgovina Unity Asset Store [25] za orodje Unity ponuja številne programske pakete. Nekateri so uporabniku na voljo brezplačno, medtem ko so drugi plačljivi. Programske pakete lahko enostavno naložimo v svoj projekt preko okolja Unity. Programski paketi, ki so na voljo v tej trgovini, so namenjeni prav razvoju aplikacij in iger v okolju Unity. Poleg tega pa spletna trgovina omogoča razvijalcem, da svoje izdelke tudi sami ponudijo drugim uporabnikom orodja Unity.

Trenutno je na tej spletni trgovini na voljo preko 30.000 produktov, med drugim so to 3D modeli, animacije, aplikacije, zvočne datoteke, celotni projekti, razširitve urejevalnika (angl. Editor extensions), sistemi delcev, skripte, senčilniki, teksture in materiali.

V diplomskem delu smo uporabili dva produkta, in sicer Graph maker in iOS Music, ki pa sta zaradi svoje pomembnosti za razvoj naše aplikacije opisana v podpoglavjih 2.1.6 in 2.1.7.

2.1.6 Graph Maker

Ker je bil namen naše aplikacije v spodbujanju uporabnika k rednemu izvajanju dihalnih vaj, smo se odločili, da bomo uporabniku znotraj aplikacije

prikazovali njegov napredek povprečnega in najdaljšega izdiha za posamezni dan.

V spletni trgovini Unity Asset Store smo imeli na voljo dve orodji za prikazovanje grafov uporabniku, in sicer GraphMaster [9] (cena 35,00 $) in Graph Maker [8] (cena 45,00 $).

Odločili smo se za uporabo orodja Graph Maker, saj ima v primerjavi z orodjem Graph Master na voljo več vrst grafov, podprto pa ima tudi nemoteno prikazovanje podatkov, v primeru da okolje Unity skalira grafične komponente glede na velikost zaslona, česar Graph Master trenutno še ne podpira. To je pomembno zaradi različnih dimenzij zaslonov mobilnih naprav. Poleg tega Graph Maker omogoča enostavno spreminjanje vseh komponent grafa, omogoča pa tudi spreminjanje podatkov med njihovim prikazovanjem.

Zaradi navedenih lastnosti se je Graph Maker za uporabo v naši aplikaciji izkazal kot idealno orodje za prikazovanje povprečnega in najboljšega izdiha v določeni časovni periodi na mobilnih napravah.

2.1.7 iOS Music

Operacijski sistem iOS je znan po svoji zaprtosti. V njem tako ni možno pridobiti pravic za brskanje po spominu celotne naprave. Za razliko od operacijskega sistema Android v operacijskem sistemu iOS ni možno na enak način poiskati željenih zvočnih datotek za predvajanje, zato smo morali dostop do zvočnih datotek znotraj naše aplikacije omogočiti na drugačen način. S tem smo uporabniku naše aplikacije omogočili izbiranje in poslušanje datotek po njegovi lastni izbiri.

V spletni trgovini Unity Asset Store [25] smo imeli na voljo dve orodji za dostopanje do zvočnih datotek na platformi iOS (angl. Music library access), in sicer iOSMusic [13] (cena 20,00 $) in iOS Music Library Access [14] (cena 30,00 $).

Odločili smo se za uporabo iOSMusic [13], ki omogoča izbiranje in predvajanje zvočnih datotek iz glasbene knjižnice na iOS napravah, omogoča pa tudi integracijo z Apple Music.

Orodje iOSMusic vsebuje vnaprej izdelano komponento (angl. Prefab [20]), prav tako imenovano iOSMusic, ki omogoča naslednje funkcije:

- **LoadAudioClip()**, ki uporabniku omogoča izbiro ene izmed njegovih zvočnih datotek ali pa izbiro enega izmed njegovih seznamov predvajanja, ki ju naloži v komponento AudioSource [26],

- **musicManager.nextSong()**, ki v AudioSource naloži naslednjo zvočno datoteko iz seznama in

- **musicManager.previousSong()**, ki v AudioSource naloži prejšnjo zvočno datoteko iz seznama.

Zgoraj opisane funkcije nam zaradi možnosti uporabe komponente AudioSource orodja Unity omogočijo enako delo z naloženimi zvočnimi datotekami, kot je to mogoče na platformi Android.

2.2 Android

Android je odprtokodni operacijski sistem, zato je zelo popularen med razvijalci in uporabniki. Omogoča namreč cenejše in lažje razvijanje programov, ki so ravno zaradi tega večinoma brezplačni [3].

Aplikacije, izdelane za Android, so v veliki večini narejene v programskem jeziku Java, z androidovim razvojnim orodjem (angl. Android software development kit), ki vključuje mnogo orodij, ki lajšajo razvoj. Najbolj znani med njimi so na primer razhroščevalnik, knjižnjice, emulatorji in primeri programske kode z navodili (angl. Tutorials) [2].

V okviru diplomskega dela smo uporabljali razvojno okolje Android Studio, verzije 2.1.2, s pomočjo katerega smo naredili knjižnjico [1], ki smo jo uporabili znotraj aplikacije. Izdelano knjižnico smo natančneje opisali v poglavju 3.4.

2.3 XML

XML je označevalni jezik, ki definira množico pravil za kodiranje dokumentov v formatu, ki je berljiv tako ljudem kot tudi strojem. Cilj oblikovanja jezika XML je bil zagotoviti njegovo enostavnost, splošnost uporabe in uporabnost na medmrežju. Gre za tekstovni podatkovni format z močno podporo standarda Unicode za uporabo različnih človeških jezikov.

Unicode je standard za konsistentno kodiranje, predstavljanje in ravnanje s tekstom v večini svetovnih pisalnih sistemov [24]. Obstaja veliko formatov dokumentov, ki uporabljajo XML sintakso, kot npr. RSS, SOAP, XHTML, prav tako ga uporablja veliko orodij, na primer Microsoft Office (Office Open XML) in OpenOffice.org. Aplikacije v Microsoft .NET razvojnem okolju pa XML datoteke uporabljajo za konfiguracijo okolja.

XML dokument je sestavljen predvsem iz značk, elementov in atributov. Značka je konstrukt, ki se začne z "<" in konča z ">". Imamo začetno značko "<section>", končno značko "</section>" in prazno značko "<linebreak/>". Element je logična komponenta v dokumentu, ki se začne z začetno značko in konča s končno, lahko pa je zgolj prazna značka. Znaki med začetno in končno značko so lahko vsebina elementa, lahko pa vsebujejo tudi druge elemente, ki jim rečemo otroci. Atribut je označevalni konstrukt, ki je sestavljen iz para "ime-vrednost" in je del začetne ali prazne značke [34].

V naši aplikaciji smo zaradi navedenih lastnosti uporabili XML označevalni jezik za beleženje podatkov o uporabniku.

2.4 MySQL

MySQL je hitra, robustna in odprtokodna rešitev za upravljanje z relacijskimi podatkovnimi bazami z jezikom SQL, ki je standarden jezik za poizvedovanje po podatkovnih bazah. Podatkovna baza omogoča efektivno shranjevanje, iskanje, sortiranje in pridobivanje podatkov. Strežnik MySQL nadzira dostop do podatkov in zagotavlja, da lahko do njih dostopajo le pooblaščeni upo-

rabniki. Najbližjo konkurenco MySQL predstavljajo PostgreSQL, Microsoft SQL Server in Oracle. Ena izmed dobrih lastnosti MySQL je njegova hitrost, poleg tega je tudi prenosljiv in deluje na različnih operacijskih sistemih, npr. Unix in Microsoft Windows [32].

V okviru diplomskega dela smo naredili podatkovno bazo, do katere smo dostopali preko PHP skripte. Tehnologijo PHP smo predstavili v nadaljevanju.

2.5 PHP

PHP je bila sprva kratica za Personal Home Page Tools (slo. orodja za osebno spletno stran), danes pa kratica pomeni PHP Hypertext Preprocessor.

PHP je skriptni jezik strežniške uporabe, namenjen predvsem razvoju dinamičnih spletnih vsebin, lahko pa se uporablja tudi za splošno programiranje. PHP kodo lahko vključimo znotraj HTML strani. Vsakič, ko obiskovalec obišče spletno stran, bo PHP koda izvedena. Napisana PHP koda je interpretirana na strežniku in generira HTML oz. drug izpis, ki bo viden obiskovalcu. PHP je odprtokodni produkt s prostim dostopom do izvirne kode, ki ga lahko uporabljamo, spreminjamo ali redistribuiramo brez stroškov [32].

V okviru diplomskega dela smo v jeziku PHP napisali skripto, s pomočjo katere se povežemo, beremo in pišemo v podatkovno bazo, ki smo jo ustvarili in se odziva na zahtevke tipa POST.

2.6 Detekcija dihanja

Kot zadnjo, a hkrati eno od pomembnejših tehnologij, ki smo jih uporabili v naši aplikaciji, bomo predstavili odprtokodno orodje za zaznavanje dihanja [18]. Orodje je namenjeno prav razvijalcem okolja Unity, saj lahko z njim v svoj projekt enostavno vključimo tudi funkcionalnost zaznavanja dihanja. Orodje za zaznavanje dihanja smo v naš projekt uvozili z uporabo Unity paketa, nato pa smo vnaprej izdelano komponento, poimenovano Breathing-

Detection, dodali v našo sceno.

Za delovanje detekcije dihanja se moramo ročno prijaviti na dogodke in definirati funkcije, ki se izvedejo ob vsaki spremembi stanja dihanja (torej ob vdihu in izdihu).

Poglavje 3

Izdelava aplikacije

Kot navedeno, so bili cilji naše aplikacije uporabniku ponuditi možnost izbire zvočnih datotek prisotnih na napravi in mu omogočiti njihovo predvajanje z izvajanjem dihalnih vaj preko mikrofonskega vhoda, v dveh načinih delovanja. Cilji aplikacije so bili tudi v tem, da aplikacija uporabniku prikaže rezultate dihalnih vaj v različnih pogledih (tedenski, mesečni ali trimesečni pogled) in jih shranjuje v XML datoteko ter uporabnika opozarja, naj med izvajanjem vaj pije vodo (asinhrono vodenje). Nenazadnje je bil cilj aplikacije tudi v njenem zaklepanju na serijsko številko.

S pomočjo v poglavju 2 opisanih tehnologij smo razvili celostno programsko rešitev, ki obsega:

- **krmilnik aplikacije**,
- **predvajalnik zvočnih datotek**,
- **izbiro zvočnih datotek**,
- **detekcijo slušalk**,
- **zaklepanje aplikacije**,
- **implementacijo detekcije dihanja**,
- **delo s podatki**.

Posamezne dele programske rešitve bomo opisali v nadaljevanju tega poglavja.

3.1 Krmilnik aplikacije

Za namen upravljanja aplikacije smo napisali skripto, imenovano AppController. Z njo vodimo uporabnika po naši aplikaciji s prikazovanjem besedilnih in slikovnih sporočil.

Krmilnik AppController upravlja predvajalnik zvočnih datotek (opisan v poglavju 3.2), preko katerega uporabniku omogočimo preklaplanje med zvočnimi datotekami in spreminjanje višine zvočnih datotek. AppController upravlja tudi z izbiro zvočnih datotek (opisano v poglavju 3.3), poleg tega pa v vsakem okviru (angl. Frame) s pomočjo detekcije slušalk (opisane v poglavju 3.4) preverja, ali so te prisotne in glede na to ustrezno reagira.

AppController skrbi tudi za zaklepanje aplikacije (opisano v poglavju 3.5) in prikazovanje pogleda za njeno odklepanje. Nadalje je AppController prijavljen na dogodke detekcije dihanja (opisano v poglavju 3.6), kamor smo vključili tudi shranjevanje in branje podatkov (opisani v poglavju 3.7). Del dogodkov detekcije dihanja je tudi predvajalnik zvočnih datotek.

3.2 Predvajalnik zvočnih datotek

Za predvajanje zvočnih datotek, ki so shranjene na mobilni napravi, smo napisali skripto, imenovano SoundPlayerScript, ki vsebuje vnaprej izdelano komponento iOSMusic, opisano v poglavju 2.1.7. Z njo upravljamo zvočne datoteke na sistemih iOS. Na sistemih Android zvočne datoteke upravljamo neposredno s komponento AudioSource.

Znotraj skripte smo napisali več funkcij. Pomembnejši med njimi sta naslednji:

- **Public void playNextSongIfEnded()**, s pomočjo katere izvemo, ali se je zvočna datoteka predvajala do konca. To za platformi Android

in iOS preverimo na enak način, le z drugo AudioSource komponento. V primeru, da se je predvajana zvočna datoteka končala, navedena funkcija poskrbi za klic druge funkcije, ki prične z nalaganjem naslednje zvočne datoteke;

- **Public float getPercentOfTime()**, ki vrača trenuten čas zvočne datoteke v vrednosti od 0 do 1, s pomočjo katere nato določimo pozicijo ročice (angl. Handle) na drsniku. Slednja uporabniku pove, kako daleč je pri poslušanju trenutne zvočne datoteke. Uporabnik namreč lahko ročico na drsniku z dotikom tudi spreminja, za kar poskrbi druga funkcija.

Poleg zgoraj opisanih funkcij v to skripto sodijo tudi funkcije, ki poskrbijo za nalaganje naslednje oziroma predhodne zvočne datoteke in jo pripravijo za predvajanje. Poleg tega omogočajo spreminjanje časa zvočne datoteke, njihovo predvajanje in ustavljanje predvajanja ter spreminjanje višine zvočne datoteke. Slednjo potrebujemo za načina delovanja, imenovana "NORMAL"in "TURBO". Zadnja funkcija, ki sodi v to skripto, služi izbiri zvočnih datotek za njihovo predvajanje, kot smo razložili v podpoglavju 3.3.

3.3 Izbira zvočnih datotek

Za izbiro zvočnih datotek smo znotraj okolja Unity napisali skripto, imenovano ChooseFolder, ki se po načinu delovanja razlikuje glede na operacijski sistem mobilne naprave, kot je opisano v nadaljevanju.

3.3.1 Android

Za namen preiskovanja datotek naprave z operacijskim sistemom Android smo razvili skripto, ki prične preiskovanje v korenskem direktoriju in hodi globje po datotečnem sistemu naprave z rekurzijo. Na ta način skripta išče poti do datotek tipa mp3 ter jih hrani v seznamu. Ko skripta preišče celotno

napravo, ustvari seznam poti do direktorijev, kjer ima uporabnik shranjene datoteke tipa mp3.

Poti direktorijev uporabniku prikažemo s pomočjo druge skripte, in sicer tako, da mu celotno pot direktorija prikažemo v obliki besedila na pripadajočem gumbu. Uporabnik s klikom na gumb izbere direktorij z datotekami tipa mp3, ki jih želi predvajati s pomočjo naše aplikacije.

Ker obstaja možnost, da je besedilo poti do zvočnih datotek na gumbih predolgo, smo se odločili, da ga s pomočjo komponente Scrollbar [22] v času premikamo od leve proti desni in obratno, ter s tem uporabniku prikažemo celotno besedilo na gumbu.

3.3.2 iOS

Za operacijski sistem iOS smo uporabili paket iOS Music (opisan v poglavju 2.1.7), s čimer smo uporabniku omogočili, da izbere zvočno datoteko oziroma seznam predvajanja zvočnih datotek. Kot pri operacijskem sistemu Android, smo tudi v tem primeru uporabili skripto, s katero smo klicali funkcijo LoadAudioClip(). Ta odpre privzeto aplikacijo za predvajanje zvočnih datotek na iOS sistemu.

Ko uporabnik izbere datoteko, se privzeta aplikacija za predvajanje zvočnih datotek zapre, funkcija LoadAudioClip pa v komponento AudioSource naloži izbrano datoteko, ki je prav tako del vnaprej izdelane komponente, ki smo jo vključili v naš projekt.

Upravljanje z zvočnimi datotekami je na operacijskem sistemu iOS enako kot na operacijskem sistemu Android.

3.4 Detekcija slušalk

Za pravilno uporabo aplikacije mora imeti uporabnik v svojo mobilno napravo vključene slušalke z vgrajenim mikrofonom, zato smo v okolju Unity poskusili implementirati detekcijo teh. Glede na dokumentacijo okolja Unity [28] je možno dobiti seznam vseh mikrofonov, ki so trenutno prisotni v na-

pravi. To smo preizkusili na platformi Android, vendar smo kot rezultat klica funkcije zaznali prisotnost le enega mikrofona, in sicer tistega, ki je del same mobilne naprave, čeprav smo imeli v mobilno napravo vključene tudi slušalke z vgrajenim mikrofonom. Na operacijskem sistemu Android smo uporabniku zato omogočili uporabo aplikacije zgolj v primeru, ko ima v mobilno napravo vključene slušalke z vgrajenim mikrofonom (opisano v 3.4.1).

Za operacijski sistem iOS te funkcionalnosti nismo implementirali, smo pa na Unity Asset Store našli programsko rešitev, s katero bi lahko na tak način nadgradili aplikacijo tudi na platformi iOS (opisano v poglavju 3.4.2).

V nadaljevanju bomo predstavili implementacijo detekcije slušalk na operacijskem sistemu Android, nato pa bomo predstavili že obstoječ programski paket z enako funkcionalnostjo še za operacijski sistem iOS.

3.4.1 Android

Pri razvoju knjižnice smo za detekcijo slušalk uporabili okolje Android Studio ter programski jezik Java. Ustvarjen razred vsebuje metode, s pomočjo katerih inicializiramo instanco razreda, nastavimo kontekst in nov razred, ki razširja že obstoječi razred BroadcastReceiver. Znotraj ustvarjenega razreda preverjamo, ali je na napravi prišlo do spremembe stanja slušalk, ki ga shranimo v spremenljivko [10].

Narejeno knjižnico smo izvozili v obliki arhiva Android knjižnice (angl. Android archive library) oz. aar ter jo nato uvozili v okolje Unity.

Okolje Unity ima obstoječa razreda, imenovana AndroidJavaClass in AndroidJavaObject, ki omogočata klicanje metod znotraj vtičnikov (angl. Native plugin). Z njuno pomočjo kličemo tudi metode knjižnice, ki smo jo ustvarili [30].

Znotraj ustvarjene skripte z inicializacijo in pridobivanjem konteksta poskrbimo za pravilne klice metod. Ti so nujni za pravilno delovanje detekcije slušalk. Za branje stanja slušalk smo naredili novo metodo, ki kliče metodo v uvoženem vtičniku in nam pove trenutno stanje slušalk. Ta metoda je javna in jo lahko uporabimo tudi v drugih skriptah.

3.4.2 iOS

V spletni trgovini Unity Asset Store je na voljo že narejena detekcija za slušalke [11] in tako predstavlja eno izmed možnih nadgradenj izdelane aplikacije. Uporaba te programske rešitve razvijalcu omogoča enostavno dostopanje do stanja slušalk in spreminjanje delovanja aplikacije. V okviru diplomskega dela smo se zato odločili, da detekcije slušalk v sistem iOS ne vključimo.

3.5 Zaklepanje aplikacije

Pri izdelavi naše aplikacije smo se odločili, da se aplikacija po določeni časovni periodi zaklene, uporabnik pa lahko nadaljuje z njeno uporabo le, če vnese pravilno šestnajstmestno serijsko številko.

Najprej smo naredili MySQL podatkovno bazo, ki služi beleženju števila registracij določene serijske številke ter preverjanju pravilnosti njenega vnosa.

Nato smo naredili PHP skripto, s katero se povežemo na podatkovno bazo. PHP skripta se odziva na zahtevek tipa POST in v njem pričakuje parametre, kot so imai številka naprave, ime aplikacije, serijska številka in vrednost, pridobljena na podlagi zgoščevalne funkcije SHA-256. Slednjo izračunamo tako, da najprej izračunamo vrednost imena aplikacije in imai številke, nato izračunamo vnešeno šestnajstmestno vrednost z žetonom in na podlagi teh dveh vrednosti še zadnjo vrednost. Izračunana vrednost se mora ujemati s poslano vrednostjo v zahtevku tipa POST, da nadaljujemo s preverjanjem prisotnosti vnešene vrednosti v podatkovni bazi.

Če vrednost, ki jo je uporabnik vnesel, v podatkovni bazi ne obstaja, uporabnika s sporočilom obvestimo o napačni serijski številki. Dokler uporabnik ne vnese pravilne vrednosti, ne more uporabljati aplikacije. Če se vnešena vrednost ujema z vrednostjo v podatkovni bazi, v njej povečamo števec registracij za to serijsko številko in kot rezultat uporabniku prikažemo sporočilo o uspešnosti vnešene vrednosti. Uporabnik v tem primeru lahko nadaljuje z uporabo aplikacije.

Za dostop do zgoraj opisane PHP skripte smo v okolju Unity naredili C# skripto. Ta na strežnik, na katerem je PHP skripta, pošlje vse potrebne zgoraj opisane parametre s pomočjo razreda WWWForm [33].

Aplikacija za odklepanje potrebuje internetno povezavo, zato uporabniku v primeru, da nima vzpostavljene internetne povezave, sporočimo, naj jo vzpostavi. Uporabnik uspešno odklene aplikacijo, ko preko vzpostavljene internetne povezave vnese pravilno serijsko številko. Skripta v tem primeru zapiše podatek o uspešni registraciji v PlayerPrefs [17]. Od tega trenutka naprej je aplikacija na tej mobilni napravi odklenjena, vse dokler je uporabnik ne izbriše in ponovno namesti. V tem primeru bo aplikacija ponovno zahtevala vnos serijske številke, vendar je registracija možna z enako serijsko številko, kot jo je uporabnik vnesel že pred izbrisom aplikacije.

3.6 Implementacija detekcije dihanja

Implementacije detekcije dihanja smo se lotili tako, da smo uvozili Unity paket ter uporabili vnaprej izdelano komponento, imenovano BreathingDetection. Znotraj skripte, opisane v poglavju 3.1, smo se prijavili na dogodke o spremembi stanja dihanja.

Ko pride do dogodka o pričetku **izdiha**, najprej preverimo, ali je uporabnik izbral direktorij, iz katerega želi poslušati zvočne datoteke (Android) oz. ali je izbral zvočno datoteko oz. seznam predvajanja (iOS) in s tem preverimo, ali je zvočno datoteko možno predvajati. Nato glede na operacijski sistem preverimo, ali se zvočna datoteka trenutno že predvaja. Če se zvočna datoteka ne predvaja, začnemo predvajati zvočno datoteko, ki jo je izbral uporabnik. V primeru, da je v aplikaciji vključen način delovanja, poimenovan "TURBO", prilagodimo višino zvočne datoteke in zamenjamo sliko, ki opisuje stanje predvajanja. Če je aplikacija v normalnem načinu delovanja, poimenovanem "NORMAL", zvočno datoteko predvajamo brez spremenjene višine zvočne datoteke. Poleg tega si zabeležimo čas začetka izdiha.

Ob dogodku **vdiha** prav tako preverimo, ali je mogoče predvajati zvočno

datoteko, in sicer na enak način, kot smo to storili ob dogodku izdiha. Če je uporabniku uspel najdaljši dosedaj zabeležen izdih, ga o tem obvestimo. V primeru, da je v aplikaciji vključen način delovanja "TURBO", prilagodimo višino zvočne datoteke. Če je aplikacija v normalnem načinu delovanja, pa začasno prenehamo s predvajanjem zvočne datoteke. Podatek o uporabnikovem izdihu zabeležimo v dogodku vdiha, posodobimo povprečen izdih tega uporabnika in ustrezno posodobimo sporočila za vodenje uporabnika.

3.7 Delo s podatki

Za potrebe dela s podatki smo naredili skripto, imenovano handlingDataXml, ki vsebuje naslednje razrede:

- **UserData**, ki vsebuje časovni žig, uporabniško ime in dolžino izdiha;
- **UserDataContainer**, ki služi za shranjevanje podatkov vseh izdihov (ki pripadajo istemu uporabniškemu imenu) v datoteko tipa XML s pomočjo serializacije [35] in prej omenjenega razreda UserData;
- **UsersOneDayData**, ki vsebuje povprečen in najdaljši izdih in časovni žig. Uporabljamo ga za prikazovanje podatkov uporabniku z izbranim uporabniškim imenom.

Omenjena skripta ima tudi naslednje funkcije:

- Shranjuje podatke o dolžini izdiha za določeno uporabniško ime v XML datoteko. To funkcijo kličemo ob vsakem dogodku vdiha;
- Vrača seznam uporabniških imen, ki so trenutno prisotni v XML datoteki in omogoča izbiro med že obstoječimi uporabniškimi imeni;
- Vrača celotno zgodovino izdihov ali zgodovino izbranega števila dni izdihov za uporabnika z izbranim uporabniškim imenom.

3.7.1 Shranjevanje podatkov

S pomočjo skripte handlingDataXml podatke uporabnika shranjujemo v lokalno XML datoteko. Pri vsakem podatku shranimo dolžino trajanja izdiha, časovni žig ter "uporabniško ime", ki si ga je izbral uporabnik. V primeru, da si ga ni izbral, je uporabniško ime "Unknown user"oz. "Neznan uporabnik". Uporabnik aplikacije lahko kadarkoli vnese tudi novo uporabniško ime in ga izbere ter nemudoma prične z dihalnimi vajami.

Podatke shranjujemo ob vsakem dogodku vdiha, ki je del detekcije dihanja, opisane v poglavju 3.6. Na tak način smo v primeru nepravilnega delovanja aplikacije ali mobilne naprave zmanjšali možnost izgube podatkov o dihalnih vajah.

3.7.2 Prikazovanje podatkov

Podatke o uporabnikovih izdihih dobimo s pomočjo skripte handleXmlData, nato pa jih v zanki dodajamo kot točke, kjer so koordinate x odvisne od tega, za kateri dan gre, koordinate y pa od uporabnikovih rezultatov. Grafu določimo njegovo največjo vrednost, ki je enaka skupnemu najdaljšemu izdihu uporabnika na koordinati y. Na koordinati x za največjo vrednost določimo število dni. Uporabnik lahko izbere enega od treh različnih časovnih pogledov podatkov, in sicer lahko izbira med enotedenskim, enomesečnim ali trimesečnim pogledom. Od izbire časovnega pogleda je odvisno tudi besedilo pripadajoče labele.

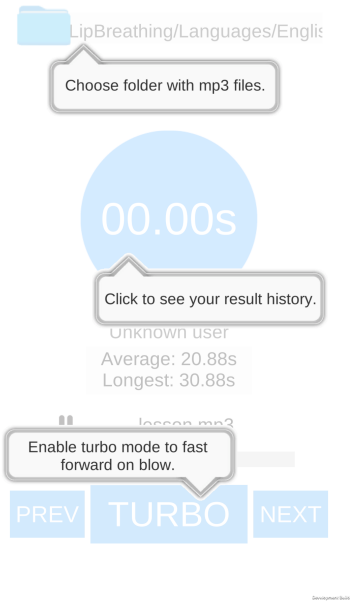
Uporabniku podatke prikažemo s črtnim grafom. Za tovrstni graf smo se odločili zato, ker bomo uporabniku za vsak dan posebej prikazali njegov povprečni in najdaljši izdih v izbranem časovnem obdobju (teden, mesec, trije meseci). Če uporabnik kakšen dan dihalnih vaj ni izvajal, sta najboljši in povprečni izdih za tisti dan enaka vrednosti 0.

Poglavje 4

Uporabnikova pot

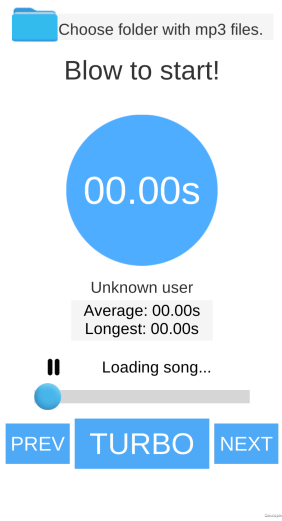
V tem poglavju bomo opisali uporabnikovo pot od namestitve aplikacije, njene uporabe, pa vse do njenega zaprtja. Osnovne informacije o uporabi aplikacije uporabniku sporočamo tako besedilno kakor tudi slikovno.

Uporabnik ob prvem zagonu aplikacije najprej vidi navodila za uporabo aplikacije, kot izhaja iz slike 4.1.



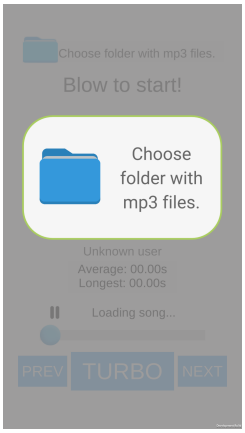
Slika 4.1: Slika začetnega zaslona z namigi za uporabnika.

Ob pritisku kamorkoli na zaslon namig za uporabo izgine in uporabnik vidi začetni zaslon, kot prikazuje slika 4.2.



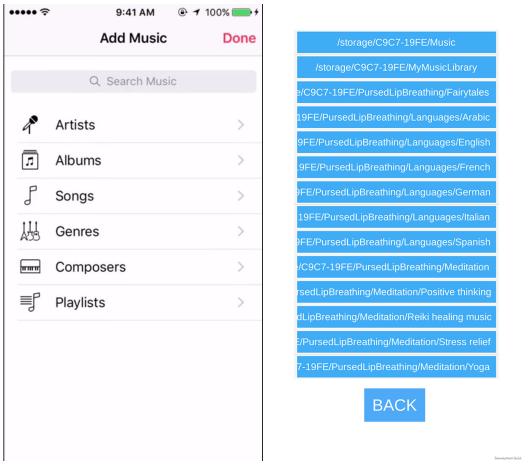
Slika 4.2: Slika začetnega zaslona aplikacije.

Preden uporabnik lahko začne z izvajanjem dihalnih vaj, mora izbrati zvočno datoteko za predvajanje. Če uporabnik prične z dihalno vadbo, preden izbere direktorij oz. zvočno datoteko za predvajanje, mu sporočimo, naj to izbere, kot je prikazano na sliki 4.3.



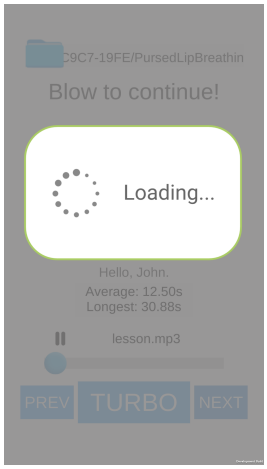
Slika 4.3: Sporočilo uporabniku, naj izbere zvočno datoteko.

Ob pritisku na sliko direktorija ali besedila ob njem uporabniku odpremo nov pogled. Na operacijskem sistemu Android se odpre pogled, ki smo ga razvili sami, na operacijskem sistemu iOS pa se odpre privzeta aplikacija, kot je razvidno iz slike 4.4.



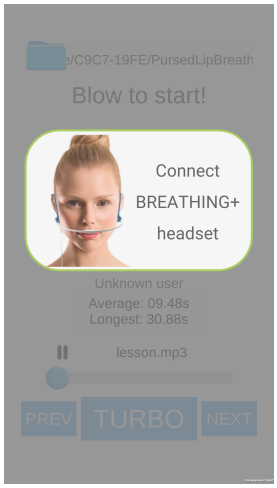
Slika 4.4: Levo operacijski sistem iOS, desno operacijski sistem Android.

Ob izbiri nove zvočne datoteke, menjavi zvočnih datotek, registraciji ipd. uporabnika obvestimo o nalaganju, kot je prikazano na sliki 4.5.



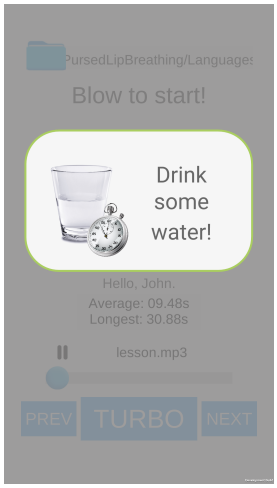
Slika 4.5: Slika ob nalaganju.

Če uporabnik kadarkoli med uporabo aplikacije iz naprave izključi slušalke z vgrajenim mikrofonom mu prikažemo sporočilo, naj vključi slušalke z mikrofonom (Android), kot je prikazano na sliki 4.6.



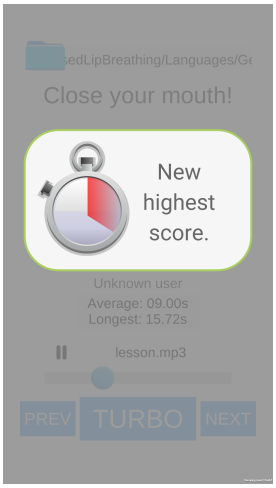
Slika 4.6: Sporočilo uporabniku, naj vključi slušalke.

Vsakih deset minut uporabniku prikažemo sporočilo, naj spije nekaj vode, kot je prikazano na sliki 4.7.



Slika 4.7: Slika sporočila uporabniku, naj spije nekaj vode.

Ko uporabnik uspe izboljšati svoj najdaljši zabeležen izdih, mu to sporočimo z obvestilom, prikazanim na sliki 4.8.



Slika 4.8: Slika sporočila o doseženem najdaljšem izdihu.

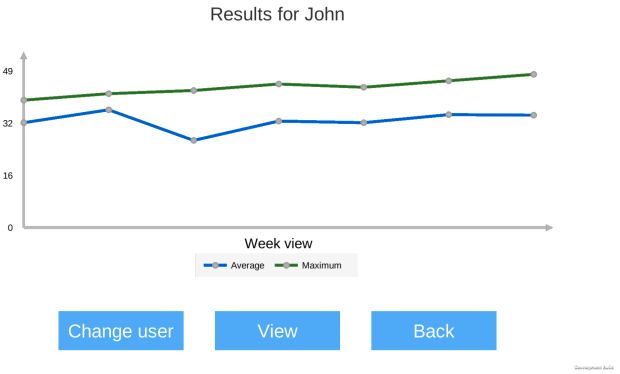
Uporabnik lahko izbira med dvema načinoma delovanja. V privzetem načinu delovanja, poimenovanem "NORMAL", se zvočna datoteka predvaja, vse dokler uporabnik piha v mikrofon in se ustavi, ko uporabnik s pihanjem preneha. S pritiskom na gumb lahko uporabnik izbere tudi poseben način delovanja, poimenovan "TURBO". V tem primeru se zvočna datoteka predvaja hitreje, in sicer z dvakratno hitrostjo, vse dokler uporabnik piha v mikrofon. Ko uporabnik s pihanjem preneha, se zvočna datoteka predvaja z normalno hitrostjo.

Pri izvajanju dihalnih vaj uporabnika vodimo tudi s prikazovanjem besedilnih sporočil.

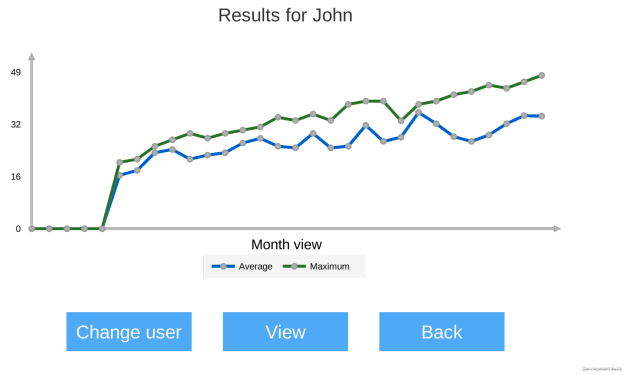
Najprej mu sporočimo, naj prične z izdihom. Ves čas trajanja izdiha mu tudi izpisujemo besedilo, naj še naprej drži izdih. Ko z izdihom preneha, mu sporočimo, da naj zapre usta. Čez eno sekundo mu prikažemo obvestilo, naj vdihne skozi nos. Uporabniku nato ponovno prikažemo sporočilo, naj prične z izdihom, kot je razvidno iz slike 4.9.

Slika 4.9: Različna sporočila uporabniku.

Ob pritisku na krog na sredini aplikacije se uporabniku odpre nov pogled. Tu lahko doda svoje uporabniško ime za shranjevanje podatkov oziroma zamenja trenutnega uporabnika. V tem pogledu si lahko ogleda tudi svoj napredek pri izvajanju dihalnih vaj, kar mu prikažemo na grafu. Izbira lahko med pogledom za tri različna časovna obdobja, in sicer za pretekli teden, kot izhaja iz slike 4.10, pretekli mesec, kot izhaja iz slike 4.11, ali za pretekle tri mesece. Uporabniku na grafu prikažemo njegov povprečni in najdaljši dih za vsak dan za izbrano časovno obdobje.

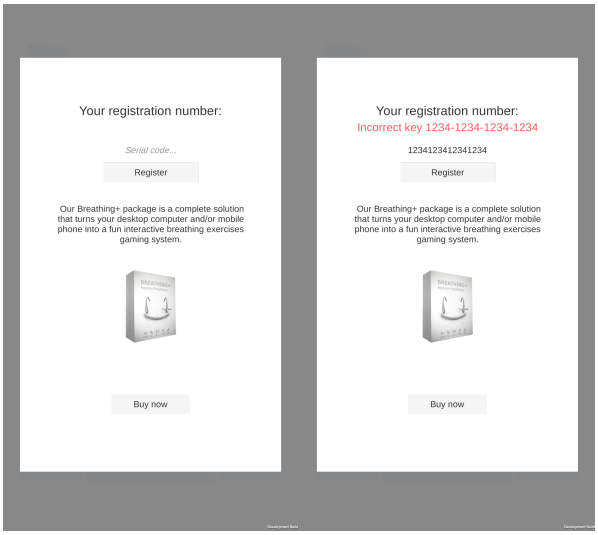


Slika 4.10: Prikaz podatkov uporabnika za pretekli teden.



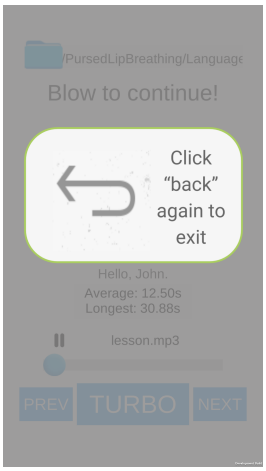
Slika 4.11: Prikaz podatkov uporabnika za pretekli mesec.

Po vnaprej določenem času ene minute od zagona aplikacije mora uporabnik vnesti šestnajstmestno številko, da z njo registrira aplikacijo. V primeru vnosa napačne številke mu to sporočimo z besedilom, kot je vidno na sliki 4.12. Če uporabnik vnese pravilno šestnajstmestno številko, se prikazano sporočilo zapre in uporabnik lahko nemoteno nadaljuje z uporabo aplikacije.



Slika 4.12: Levo slika pogleda za registracijo naprave, desno slika obvestila o napačnem vnosu.

Uporabnik aplikacijo zapre z dvakratnim zaporednim klikom gumba "nazaj"v dovolj kratkem časovnem intervalu. To uporabniku sporočimo ob prvem pritisku gumba nazaj, kot je vidno na sliki 4.13.



Slika 4.13: Prikaz izhoda iz aplikacije.

Poglavje 5

Diskusija

V okviru diplomskega dela smo razvili mobilno aplikacijo za predvajanje zvočnih datotek z nadzorom preko mikrofonskega vhoda v okolju Unity za operacijska sistema Android in iOS. Namen izdelane aplikacije je izvajanje dihalnih vaj s tehniko dihanja skozi priprte ustnice, imenovano PLB, s katero skušamo uporabniku pomagati pri lajšanju nekaterih (tudi zdravstvenih) težav. Cilj izdelane aplikacije je bil uporabniku ponuditi zabavno in zanimivo aplikacijo, ki bo poleg tega tudi pozitivno vplivala na njegovo zdravje in počutje.

Aplikacijo smo zasnovali tako, da uporabnika motivira pri izvajanju dihalnih vaj v daljšem časovnem obdobju, saj ga med njihovim izvajanjem vodi in spodbuja ter mu omogoča, da svoj napredek spremlja v izbranem časovnem obdobju.

Zastavljene cilje je naša aplikacija dosegla na naslednje načine:

- Aplikacija uporabniku ponudi izbiro zvočnih datotek, prisotnih na napravi. Na operacijskem sistemu Android aplikacija ob zagonu preišče vse zvočne datoteke na napravi in jih uporabniku ponudi v pogledu, ki smo ga razvili sami, kot je razvidno iz slike 4.4. Če uporabnik izbere novo zvočno datoteko, aplikacija prične z nalaganjem te, kot je prikazano na sliki 4.5. Na platformi iOS uporabniku ponudimo izbiro zvočne datoteke s pomočjo paketa iOS Music, opisanega v podpoglavju 3.3.2;

- Aplikacija predvaja zvočne datoteke z izvajanjem dihalnih vaj preko mikrofonskega vhoda in ima dva načina delovanja, poimenovana "NORMAL"in "TURBO", opisana v poglavju 4. Izdelana aplikacija se odziva v odvisnosti od vhoda na mikrofonu v manj kot 400ms. Večina že obstoječih aplikacij, ki uporabljajo mikrofonski vhod, ga običajno uporablja za prepoznavanje govora (npr. uporaba Google Cloud Speech API), kar v aplikacije prinaša mnogo večjo latenco (približno 1-2 sekundi);

- Aplikacija ob vsakem dogodku vdiha sproti shranjuje podatke v obliki XML datoteke. Shranjene podatke aplikacija prikazuje v treh različnih pogledih, in sicer omogoča tedenski (vidno na sliki 4.10), mesečni (vidno na sliki 4.11) in tromesečni pogled. Poleg tega aplikacija omogoča tudi dodajanje novih uporabnikov in tako možnost vodenja evidence več uporabnikov na isti napravi;

- Med uporabo aplikacija uporabnika vsakih deset minut opozori, naj pije vodo, kot je prikazano na sliki 4.7. S tem preprečimo dehidracijo, do katere lahko prihaja zaradi izvajanja dihalne vadbe;

- Aplikacija se po vnaprej določeni časovni periodi zaklene, uporabnik pa lahko z njeno uporabo nadaljuje le, če vnese pravilno šestnajstmestno serijsko številko, kot je prikazano na sliki 4.12.

Zaradi razlik med operacijskima sistemoma iOS in Android smo v postopku izdelave aplikacije kljub uporabi orodja Unity naleteli na nekatere težave pri preiskovanju naprav s sistemom iOS. V postopku izdelave aplikacije smo za omenjeno težavo našli ustrezno rešitev, ki smo jo opisali v poglavju 3.3.2.

Naslednja večja težava, ki se je pri izdelavi aplikacije še pojavila, je bila povezana z detekcijo slušalk z vgrajenim mikrofonom. Za naprave s sistemom Android smo izdelali ustrezno programsko rešitev, opisano v poglavju 3.4.1, za naprave s sistemom iOS pa smo predstavili možno rešitev v obliki nadgradnje naše aplikacije, opisano v poglavju 3.4.2.

Izdelana aplikacija na trgu predstavlja novost na področju dihalnih vaj s tehniko dihanja skozi priprte ustnice. Uporabo naše aplikacije priporočamo vsem, ki želijo na zanimiv in uporabniku prijazen način lajšati navedene zdravstvene težave današnjega vsakdana v udobju svojega doma.

Literatura

[1] Android library. Dosegljivo: https://developer.android.com/studio/projects/android-library.html. [Online; Dostopano: 11. 9. 2016].

[2] Android. Dosegljivo: https://en.wikipedia.org/wiki/Android_software_development. [Online; Dostopano: 26. 9. 2016].

[3] Operacijski sistem Android. Dosegljivo: https://sl.wikipedia.org/wiki/Android_(operacijski_sistem). [Online; Dostopano: 26. 9. 2016].

[4] Aplikacija - Bowling. Dosegljivo: https://play.google.com/store/apps/details?id=air.com.breathinglabs.BreathingGames.Bowling. [Online; Dostopano: 24. 10. 2016].

[5] Aplikacija - Breathe2Relax. Dosegljivo: http://t2health.dcoe.mil/apps/breathe2relax. [Online; Dostopano: 24. 10. 2016].

[6] Aplikacija - EZair. Dosegljivo: http://bfe.org/new/try-our-breath-pacer-ez-air-plus/. [Online; Dostopano: 24. 10. 2016].

[7] Unity - Forum. Dosegljivo: http://forum.unity3d.com/threads/audio-auto-disabled-after-any-system-sound-on-android.405849/. [Online; Dostopano: 10. 9. 2016].

[8] Graph Maker - Asset Store. Dosegljivo: https://www.assetstore.unity3d.com/en/#!/content/11782. [Online; Dostopano: 14. 9. 2016].

[9] Graph Master - Asset Store. Dosegljivo: https://www.assetstore.unity3d.com/en/#!/content/12637. [Online; Dostopano: 14. 9. 2016].

[10] Action Headset Plug - Android Developer. Dosegljivo: https://developer.android.com/reference/android/media/AudioManager.html#ACTION_HEADSET_PLUG. [Online; Dostopano: 26. 9. 2016].

[11] Headphone Detector - Asset Store. Dosegljivo: https://www.assetstore.unity3d.com/en/#!/content/28626. [Online; Dostopano: 26. 9. 2016].

[12] Unity - Manual: Interface. Dosegljivo: https://docs.unity3d.com/Manual/LearningtheInterface.html. [Online; Dostopano: 25. 9. 2016].

[13] iOS Music - Asset Store. Dosegljivo: https://www.assetstore.unity3d.com/en/#!/content/31128. [Online; Dostopano: 14. 9. 2016].

[14] iOS Music Library Access - Asset Store. Dosegljivo: https://www.assetstore.unity3d.com/en/#!/content/30258. [Online; Dostopano: 14. 9. 2016].

[15] Margaret A Nield, Guy W Soo Hoo, Janice M Roper, and Silverio Santiago. Efficacy of pursed-lips breathing: a breathing pattern retraining strategy for dyspnea reduction. *Journal of cardiopulmonary rehabilitation and prevention*, 27(4):237–244, 2007.

[16] Aplikacija - Pacedbreathing. Dosegljivo: http://pacedbreathing.blogspot.si/. [Online; Dostopano: 24. 10. 2016].

[17] Unity - Scripting API: PlayerPrefs. Dosegljivo: https://docs.unity3d.com/ScriptReference/PlayerPrefs.html. [Online; Dostopano: 1. 10. 2016].

[18] Tilen Pogačnik. *Ogrodje za zaznavanje dihanja in predvajanje video posnetkov v okolju Unity.* Diplomsko delo, Ljubljana: Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2016.

[19] Aplikacija - Pranayama. Dosegljivo: `http://www.saagara.com/apps/breathing/universal-breathing-pranayama`. [Online; Dostopano: 24. 10. 2016].

[20] Unity - Manual: Prefabs. Dosegljivo: `https://docs.unity3d.com/Manual/Prefabs.html`. [Online; Dostopano: 25. 9. 2016].

[21] Matthew Price, Erica K. Yuen, Elizabeth M. Goetter, James D. Herbert, Evan M. Forman, Ron Acierno, and Kenneth J. Ruggiero. mhealth: A mechanism to deliver more accessible, more effective mental health care. *Clinical Psychology & Psychotherapy*, 21(5):427–436, 2014.

[22] Unity - Manual: Scrollbar. Dosegljivo: `https://docs.unity3d.com/Manual/script-Scrollbar.html`. [Online; Dostopano: 1. 10. 2016].

[23] Aplikacija - Swimmer. Dosegljivo: `https://play.google.com/store/apps/details?id=com.BreathingLabs.Swimmer`. [Online; Dostopano: 24. 10. 2016].

[24] Unicode. Dosegljivo: `https://en.wikipedia.org/wiki/Unicode`. [Online; Dostopano: 11. 9. 2016].

[25] Unity Asset Store. Dosegljivo: `https://www.assetstore.unity3d.com/`. [Online; Dostopano: 12. 9. 2016].

[26] Unity - Scripting API: AudioSource. Dosegljivo: `https://docs.unity3d.com/ScriptReference/AudioSource.html`. [Online; Dostopano: 1. 10. 2016].

[27] Unity Game Engine. Dosegljivo: `https://en.wikipedia.org/wiki/Unity_(game_engine)`. [Online; Dostopano: 11. 9. 2016].

[28] Unity - Scripting API: Microphone. Dosegljivo: https://docs.unity3d.com/ScriptReference/Microphone-devices.html. [Online; Dostopano: 26. 9. 2016].

[29] Unity - Multiplatforms. Dosegljivo: https://unity3d.com/unity/multiplatform. [Online; Dostopano: 19. 9. 2016].

[30] Unity - Manual: Plugin for android. Dosegljivo: https://docs.unity3d.com/Manual/PluginsForAndroid.html. [Online; Dostopano: 11. 9. 2016].

[31] Unity - Store. Dosegljivo: https://store.unity.com/. [Online; Dostopano: 19. 9. 2016].

[32] Luke Welling and Laura Thomson. *PHP and MySQL Web development*. Sams Publishing, 2003.

[33] Unity - Scripting API: WWWForm. Dosegljivo: https://docs.unity3d.com/ScriptReference/WWWForm.html. [Online; Dostopano: 11. 9. 2016].

[34] XML. Dosegljivo: https://en.wikipedia.org/wiki/XML. [Online; Dostopano: 11. 9. 2016].

[35] Microsoft .NET - Examples of XML Serialization. Dosegljivo: https://msdn.microsoft.com/en-us/library/58a18dwa(v=vs.110).aspx. [Online; Dostopano: 11. 9. 2016].